Pedestrian Motion Prediction Using Deep Learning

Pedro Miguel Gustavo Bilro
Instituto Superior Técnico, ULisboa,
Lisboa, Portugal, 1049–001
Email: pedro.bilro@tecnico.ulisboa.pt

***Abstract*—Pedestrian motion prediction is a task that is relevant for many kinds of intelligent systems. However, it can be quite challenging, due to the fact that humans can be influenced by a plethora of factors. In recent years, two types of factors have been getting more relevance: the presence of obstacles or social interactions. Most methods that incorporate both of these types require information like video frames or semantic maps, which may not be readily available. We propose a new model, named Arc-LSTM-SMF, which considers the existence of obstacles, as well as social interactions, using only pedestrian trajectories. This model integrates Sparse Motion Fields with Long Short Term Memory networks, with the use of a new pooling layer that simulates a field of view for each pedestrian. We evaluate our model using standard geometric metrics, as well as metrics related to obstacle avoidance and pedestrian collision avoidance. The proposed Arc-LSTM-SMF is able to outperform several state-of-the-art models on popular pedestrian datasets. The model is open-source and is available at https://github.com/pedro-mgb/pedestrian-arc-lstm-smf.**

Keywords: Machine learning; Trajectory prediction; Social interactions; Obstacle awareness

1. Introduction

Knowing how pedestrians move and interact with their surrounding environment, which can include pedestrians and other obstacles, is a crucial process for many kinds of intelligent systems. Service robots and autonomous vehicles need to adjust their trajectories in order to coexist with humans [1], [2]. Surveillance systems may need to properly predict the motion of people in a video scene to infer what kinds of activities are being conducted [3], [4].

Pedestrian trajectory prediction can be summed up to predicting where a pedestrian will go for a certain foreseeable future. This is a very challenging task, because the motion of pedestrians can depend on a variety of factors, such as:

1) **Presence of obstacles in the scene** [5]: Pedestrians will avoid colliding with static obstacles, and will preferably go through areas were traversal is possible.
2) **Presence of other pedestrians** [6]: Pedestrians may interact with each other in many ways, which undoubtedly can influence their motion.
3) **Personal characteristics** [7]: Specific attributes like age and health can influence pedestrian motion.
4) **Individual and/or group goals** [7]: Each person can have a specific place to go to, which can mean a strict path to follow. The same can also apply for a group of people (if multiple pedestrians are considered).

Of the aforementioned factors, the ones usually considered in trajectory forecast are the presence of obstacles and the presence of other pedestrians.A visual representation of these two factors can be seen in fig. 1.

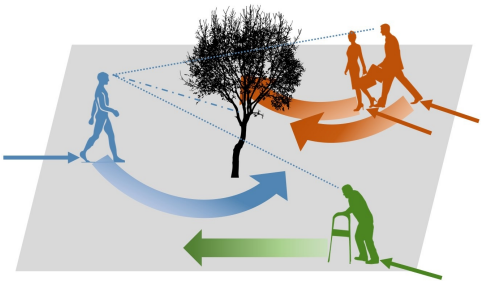


Fig. 1. Illustration of the trajectory prediction problem and its difficulty. In this example, the man in blue (left side of the figure) will adjust his trajectory to avoid colliding with a tree, and with other people also present in the scene. Similarly, a trajectory prediction method should deal with the presence of static obstacles, and of other pedestrians that can interact with each other.

Classical works consisted in physics-based models, often using handcrafted features [6], [8], [9] In recent years, there have been advancements in creating neural network architectures for trajectory forecast [10]–[15]. While some models try to incorporate interactions between pedestrians in a data-driven way [10], [11], others process scene-specific environments with specialized networks [2], [12]. There has even been a recent body of work to consider both social and scene aspects in one model [4], [13], [15].

Most of the aforementioned models suffer from limitations. Those that only consider scene or social aspects will have limited performance in situations where both aspects have weight. While the current models that integrate these two aspects are meant to combat such limitations, they usually require extra information. This extra information may be in the form of semantic maps or video frames, which may not always be readily available [16], [17]. Furthermore, the processing of such data requires additional networks that can be computationally heavy [18]–[20].

We propose a socially-aware and scene-compliant model that does not require video data. The model, labelled henceforth as "Arc-LSTM-SMF", will be based on Long Short Term Memory (LSTM) [21]–[23] - a type of Recurrent Neural Network (RNN) specialized in handling long-term dependencies in sequences - to sequentially process pedestrian motion. It will be combined with Sparse Motion Fields (SMF) [5], [24], a scene-specific model that learns the presence of obstacles using just pedestrian trajectories. The interactions between pedestrians will be incorporated using an improved version of directional pooling [25], that focuses on the neighbours in

front of the pedestrian. The performance of our Arc-LSTM-SMF model will be demonstrated by comparing predicted pedestrian trajectories of our model with other state-of-the-art works, using popular trajectory forecasting datasets [9], [26], and the Trajnet++ benchmark [25].

2. Related Work

Human trajectory forecasting methods can be categorized by the type of contextual cues they incorporate [7]. Static environment cues relate to the presence of obstacles and unwalkable areas and are specific to each scene. Dynamic environment cues consider the presence and interactions between people and/or groups of people. There are also pedestrian-specific cues (*e.g.*, age, health, and personal goals), but since our model does not consider these, they will not be detailed to much extent. We present existing research that incorporates the first two types of cues and discuss their breakthroughs and limitations.

Scene-specific information: There have been several lines of research into the incorporation of scene-specific elements - obstacles and traversal restrictions - for trajectory forecast. Semantic maps have proven to be a solid option [2], [12], [27], which can have the location of obstacles, but also the presence of roads and poor terrain. The CAR-NET [2] model takes as input pedestrian trajectories and a top-view image of the scene. The latter is supplied to a Convolutional Neural Network (CNN) [18] in order to extract feature maps through scene segmentation techniques [28]. Ridel et al. [12] also extracted features from the scene, but using ResNet [20]. Those features are merged with trajectory data to form a probability grid that indicates the most likely location of each pedestrian. Both the aforementioned works require feature extractors, which can make the whole model computationally heavier. Barata et al. [5] proposed a model based on SMF that would learn the presence of obstacles and areas without pedestrian traversal, using only the actual pedestrian trajectories. The motion fields restrict the learnt representation of pedestrians' motion to areas where motion is actually possible. These models do not consider interactions between pedestrians, and therefore can have limited performance in crowded scenarios.

Social interactions: This has been the most common type of relevant cue to be considered in the trajectory forecasting task [7]. The social force model [6] is physics-based, using repulsion and attraction forces between pedestrians. While it is still being used in recent works [8], [29], its use of handcrafted energy potentials makes model generalization difficult. Alahi et al. [10] were one of the first to use LSTM networks to create a data-driven trajectory forecasting method, named Social-LSTM. The hidden states of neighbouring pedestrians were combined in a procedure called social pooling. An improved version of this model, named Directional LSTM [25], used relative velocities of neighbours instead of hidden states, which are harder to interpret. Other types of networks have also been used. Gupta et al. [11] used Generative Adversarial Networks (GANs) [30], [31] combined with a pooling module to generate multiple socially acceptable trajectories. Other variants of GANs have also been employed in this task [32], [33]. Mohamed et al. [14] have obtained competitive results, modelling interactions with spatio-temporal graphs.

Scene and social context: Several models that capture scene-specific information and social interactions have been proposed in recent years [4], [13], [15], [34], [35]. Some works took as basis a social model [10] and extended it to also incorporate the presence of obstacles [36]–[38]. Others extended scene models [2] to also consider social interactions [13], [35]. The NEXT model from [4] uses bounding boxes surrounding people and objects in a video scene, together with semantic maps, and a behaviour module to incorporate social interactions. The Trajectron++ [15] is currently one of the best performing models in this task. It incorporates LSTM networks with other kinds of RNNs. It also uses CNNs for scene processing (if available). Interactions between pedestrians - and potentially other types of agents such as cyclists or skaters - are encoded as edges of a spatio-temporal graph. These methods require additional information regarding the scene to perform accurate trajectory prediction, which increases the computational weight, while also adding an additional dependency on data that may not always be available (*e.g.*, access to GPS data alone [16]). To the best of our knowledge, our model is the first to incorporate both social and scene constraints, using only trajectory data as input.

3. Method

Pedestrians tend to change their path or velocity to accommodate for other pedestrians. Furthermore, they may also do the same to avoid physical obstacles that may stand in their way. A trajectory prediction method needs to be able to predict such changes of motion. To do that, it needs to integrate, directly or indirectly, the presence of obstacles and of other pedestrians in its predictions. The proposed Arc-LSTM-SMF model, receives scene-specific predictions from the SMF method [5], and considers the presence of neighbours with an arc - or Field of View (FOV) - pooling layer. This section will describe the model in parts: scene integration, social aspects, and motion processing. All these parts are combined to form the full Arc-LSTM-SMF model.

3.1. Problem Definition

The trajectory forecasting problem can be parameterized with the notation similar to [10]. The input is a sequence of 2D trajectories for all people simultaneously present in the scene, $\mathbb{X}_i$, with $X_i^t \in \mathbb{R}^2, i \in \{1, ..., N\}, t \in \{T_{ini}, ..., T_{obs}\}$ defining the position at instant t, and N being the total number of pedestrians. The objective is to estimate a future trajectory for each of the pedestrians, $\hat{\mathbb{Y}}_i = \left\{\hat{Y}_i^{T_{obs}+1}, ..., \hat{Y}_i^{T_{pred}}\right\}$, with $\hat{Y}_i^t$ being the predicted position of pedestrian i and time $t > T_{obs}$. The prediction should be as close to the real future trajectory, $\mathbb{Y}_i = \left\{Y_i^{T_{obs}+1}, ..., Y_i^{T_{pred}}\right\}$, as possible. The real future trajectory will also be referred as Ground Truth (GT). The Arc-LSTM-SMF model works with displacements between two consecutive positions instead of actual absolute positions.

They are defined as ΔX_i^t, $\Delta \hat{Y}_i^t$, and ΔY_i^t, for past, predicted, and GT displacements.

3.2. Sparse Motion Fields

The proposed model for pedestrian motion prediction has as basis recent methods proposed for the same task. Namely, the integration of scene-specific elements are done via the SMF method. It learns the presence of unwalkable regions in an unsupervised way, *i.e.*, without access to scene information, and using just the pedestrian trajectories. A short introduction into SMF is given here. For more information on the method, refer to [5].

Motion fields are applied with the previous 2D position, outputting a 2D displacement. For each pedestrian i, the relation between two consecutive positions (instants t and $t-1$) is defined as:

$$X_i^t = X_i^{t-1} + \Gamma_{k_i^t}\left(X_i^{t-1}\right) + w_i^t, \tag{1}$$

where the current and previous positions, X_i^t and X_i^{t-1}, are replaced by the predictions, $\hat{Y}_i^t$ and $\hat{Y}_i^{t-1}$, for $t > T_{obs}$ (since there is no more input). The term $k_i^t \in \{1, \ldots, K\}$ - with K being total number of motion fields - identifies the active motion field $\Gamma_{k_i^t} : [0,1]^2 \rightarrow \mathbb{R}^2$ governing the displacement at current time step t. It is worth mentioning that the input is normalized, $X_i^t \in [0,1]^2$, where $[0,1]^2$ denotes the image lattice. There is also additive white noise, $w_i^t \sim N\left(0, \Sigma_{k^t}(X_i^t)\right)$, with the covariance matrix Σ_{k^t} depending on the 2D position of the pedestrian. The method parameters are estimated using the Expectation-Maximization (EM) algorithm [39]. The To perform inference, the most likely motion field, $\hat{k}_i^t$, is chosen using a forward pass of E-step algorithm (the first step of EM). A deterministic prediction can be done by using the motion field $\hat{k}_i^t$, and removing the noise in (1).

3.3. The Arc-LSTM-SMF model

The proposed model for scene and interaction-aware trajectory forecasting, Arc-LSTM-SMF, can be divided in three main modules:

1) LSTM networks, that are responsible for processing the pedestrian trajectories and generating the predictions of the future trajectories. In fact, the LSTM networks do not work directly with the positions, but instead with the displacement between two positions, since it has been proven that it is easier to predict relative motion instead of absolute positions [11], [14], [32]. Similarly to other works [11], [25], we adopt an encoder-decoder architecture. This means that there are two LSTMs, with the encoder processing the past trajectory, and the decoder forecasting the future trajectory.
2) SMF method, summarized in section 3-B, that generates scene-specific (and scene-compliant) predictions that are fed to the decoder when generating its own predictions. It is this integration that makes our model scene-aware.
3) An arc (or FOV) pooling layer, that takes the positions of all pedestrians at that instant. It generates a tensor containing relevant social context, which is sent to the LSTM cell (encoder or decoder). It this is layer that makes our model interaction-aware.

The overall architecture of the Arc-LSTM-SMF model is illustrated in fig. 2. It shows the aforementioned three modules and the connections between each of them.

3.4. Arc pooling layer

The consideration of social interactions in a model based on LSTMs has commonly been done by including an interaction layer. This layer receives information from multiple pedestrians and outputs a tensor with information regarding the neighbourhood each pedestrian. The first major use of such a layer was in the Social-LSTM model [10]. An alternative was proposed with Directional LSTM [25], which used the relative velocities of neighbouring pedestrians. Being easier to interpret and less computationally heavy, the latter was chosen as basis to build our own interaction layer. Directional LSTM builds a social tensor on a square grid containing $N_g \times N_g$ cells, centered on the pedestrian. All cells of the grid have length l, with each cell containing the information of the neighbours whose position lies in that cell. Neighbours outside the grid are not considered for pooling. The tensor generated by each pedestrian has size $N_g \times N_g \times 2$.

Using a square grid has an issue associated to it: the neighbours behind the pedestrian are included. While in some cases these can influence the pedestrian's motion, the pedestrian's focus usually lies on the neighbours in front, *i.e.*, the ones that are visible [40]. This motivated the following improvement to Directional LSTM: instead of considering a square grid, we employ an arc-shaped pooling to consider only the neighbours that the pedestrian can see. As such, the arc shape can be thought of as the FOV of the pedestrian.

A visualization of the two pooling techniques - grid and FOV - is available in fig. 3. As seen in fig. 3b, the grid shape considers a pedestrian (in green) that is behind and moving away from the blue pedestrian. The green pedestrian most likely does not influence the motion of the blue pedestrian, and as such does not need to be including the pooled tensor. The FOV (fig. 3c) only includes the orange pedestrian, which is the one that the blue pedestrian is most likely focusing on.

The orientation of the FOV is be given by the pedestrian's gaze direction. However, as stated in the problem definition of section 3-A, the proposed models only have access to the actual trajectories of pedestrians, in $\mathbb{R}^2$. To circumvent this, it is simulated using the direction of motion.The FOV shape can have more degrees of freedom than a square grid. In our work, the FOV has a radius r and spread (or angle) α. Furthermore, it has $N_\alpha \times N_r$ cells, with N_α N_r being the number of divisions in the spread and radius, respectively. For instance, the FOV in fig. 3c has $N_\alpha = 3$ and $N_r = 4$.

The output of the arc pooling layer, for each pedestrian i is a $N_r \times N_\alpha \times 2$ tensor containing the relative velocities (or displacements) or each neighbour that is inside the arc.

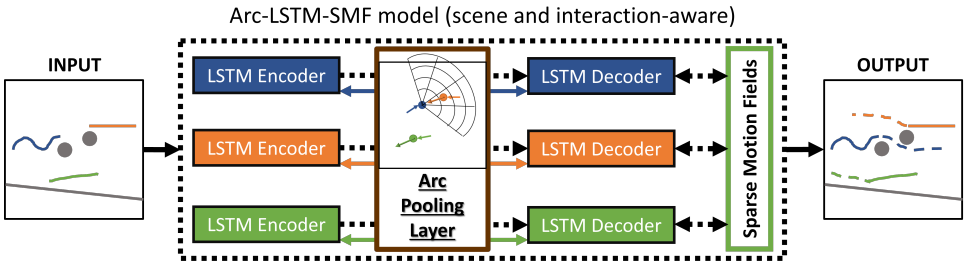
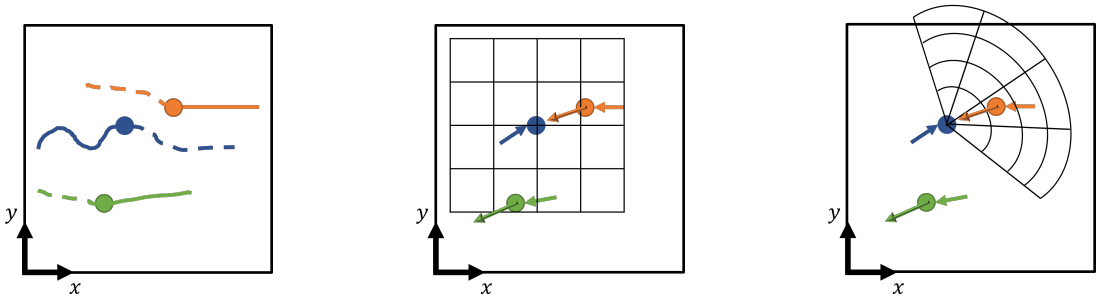


Fig. 2. Arc-LSTM-SMF architecture for scene and interaction-aware trajectory forecast. The input is a set of trajectories for all pedestrians present in a scene. The pedestrians may have obstacles in the way (in gray, not apart of input). The social context surrounding each pedestrian is obtained every instant via an arc pooling layer. The SMF predictions are sent to the LSTM decoder to have a scene-specific consideration of the environment.



(a) Full trajectories (b) Grid pooling [25] (c) FOV pooling (ours)

Fig. 3. Comparing directional pooling (originally from [25]) using different shapes. **(a)** Trajectories of 3 pedestrians, with full line representing the past and dashed representing the future. The pooling is focused on the blue pedestrian. **(b)** The grid shape considers a pedestrian (in green) that is behind and moving away from the blue pedestrian. **(c)** Our FOV based shape only includes the orange pedestrian, that the blue pedestrian is most likely focusing on.

Formally, for the (m_r, n_α) cell, the tensor is defined as:

$$A_i^t(m_r, n_\alpha, :) = \frac{1}{\sum_{j \in \mathcal{N}_i} \mathbf{1}_{m_r,n_\alpha}\left[d_{ij}^t, \beta_{ij}^t\right]} \sum_{j \in \mathcal{N}_i} \mathbf{1}_{m_r,n_\alpha}\left[d_{ij}^t, \beta_{ij}^t\right]\left(\Delta X_j^t - \Delta X_i^t\right), \quad (2)$$

where d_{ij}^t is the distance between pedestrian i and j, and β_{ij}^t is their relative orientation. The difference of displacements $\left(\Delta X_j^t - \Delta X_i^t\right)$ is used in place of the actual relative velocity. The indicator function $\mathbf{1}_{m_r,n_\alpha}[d_{ij}^t, \beta_{ij}^t]$ checks if neighbour j is inside the (m_r, n_α) cell of the FOV of pedestrian i. The first term $\left(\frac{1}{\mathbf{1}_{m_r,n_\alpha}[d_{ij}^t,\beta_{ij}^t]}\right)$ means that if there are multiple neighbours in the same cell, their relative velocity is averaged.

3.5. Integrating SMF and Arc pooling with LSTM

Having described the SMF (scene) and arc pooling (social) modules, this section details how the information that they output is integrated into the LSTM networks.

LSTM encoder. The encoder processes the past trajectory, available as input, while also receiving a tensor containing social information. Since the model receives SMF predictions, these are only used by the decoder. Both position and arc-shaped tensor, for each pedestrian i and instant t, are embedded before being fed to the LSTM cell:

$$e_i^t = Emb\left(\Delta X_i^t; W_{emb}\right), \quad (3)$$

$$p_i^t = Emb\left(A_i^t; W_{emb_A}\right), \quad (4)$$

where each Emb layer includes an affine transformation (weights and biases) and a PReLU activation function [14]. The embedding of the position, e_i^t, and of the arc-shaped tensor p_i^t, are done with separate parameters, W_{emb} and W_{emb_A}. The hidden state of the LSTM encoder, h_i^t, is computed with the previous state and the embeddings:

$$h_i^t = LSTM_e\left(h_i^{t-1}, \left[e_i^t, p_i^t\right]; W_{LSTM_e}\right), \quad (5)$$

where the last hidden state outputted by the encoder, $h_i^{T_{obs}}$, contains a encoded summary of the full past trajectory of pedestrian i. This summary is the information that is sent for the LSTM decoder.

LSTM decoder. The decoder is responsible for generating the predicted motion for each pedestrian, one instant at a time. Besides processing the position and social context of each tensor, the LSTM decoder also processes the predicted displacements from the SMF. Instead of just receiving the most likely displacement, the LSTM receives K displacements from the K different motion fields. This gives extra flexibility to the LSTM network in giving importance to the scene-specific predictions. These K predicted displacements from motion fields are also embedded:

$$q_i^t = Emb\left(\Gamma_1\left(\hat{Y}_i^{t-1}\right), ..., \Gamma_K\left(\hat{Y}_i^{t-1}\right); W_{emb_q}\right) \quad (6)$$

where $\Gamma_k\left(\hat{Y}_i^{t-1}\right)$ is the k-th motion field displacement at time t for pedestrian i. Notice the motion fields receive the previous predicted position from the LSTM, $\hat{Y}_i^{t-1}$. This is a correction done on the motion fields. Even though they

are scene-compliant, their predictions can be poor from a geometric standpoint. The LSTM decoder cell receives q_i^t, as well as the other two embeddings, to compute the hidden state:

$$h_i^t = LSTM_d\left(h_i^{t-1}, \left[e_i^t, q_i^t, p_i^t\right]; W_{LSTM_d}\right) \tag{7}$$

The first decoder step receives the state $h_i^{T_{obs}}$, and the final hidden state outputted is $h_i^{T_{pred}-1}$. The output of the model is no longer a single displacement value, but instead a probabilistic distribution, Similarly to [10], [14], [25], we use a bi-variate Gaussian distribution for the displacement, with a total of 5 parameters: 2D mean μ, 2D standard deviation σ, and correlation factor ρ. These parameters are obtained from the state, via an output linear layer:

$$\left[\hat{\mu}_i^{t+1}, \hat{\sigma}_i^{t+1}, \hat{\rho}_i^{t+1}\right] = W_{out_{lin}} h_i^t + b_{out_{lin}}, \tag{8}$$

where $W_{out_{lin}}$ and $b_{out_{lin}}$ are the weights and bias of the output layer. To obtain a deterministic prediction, the mean displacements can be used: $\Delta\hat{Y}_i^t \equiv \hat{\mu}_i^{t+1}$. The estimated parameters $\hat{\sigma}_i^{t+1}$ and $\hat{\rho}_i^{t+1}$ are used in training the model.

3.6. Training

Since the model outputs parameters of a probabilistic distribution, the training is done in a probabilistic manner. Following the procedure of recent works [10], [25], we use a Negative Log Likelihood (NLL) loss:

$$L_i\left(W_{net}\right) = -\sum_{t=T_{obs}+1}^{T_{pred}} \log\left(\mathbb{P}\left(\Delta Y_i^t \mid \hat{\mu}_i^t, \hat{\sigma}_i^t, \hat{\rho}_i^t\right)\right), \tag{9}$$

where the subscript i in L_i means the loss is computed for each pedestrian. The parameters of the Arc-LSTM-SMF, W_{net} are learned by minimizing the loss in (9).

When evaluating the model, there is no access to GT, so for $t > T_{obs}$ the predicted displacements are used as input to the model. At train time, the GT displacements are used as input to the Arc-LSTM-SMF model. This technique is often referred to as teacher forcing and has proven to help with training trajectory forecasting models [10], [25].

3.7. Implementation Details

The encoder and decoder LSTMs have the same parameter dimensions (although having separate parameters). We use a hidden state (h_i^t) dimension of 128. The embedding of pedestrian displacement (e_i^t) and SMF displacements (q_i^t) each have size 32. The FOV has angle $\alpha = 140°$ and radius $r = 4\,\mathrm{m}$, with a total of $N_r \times N_\alpha = 20$ cells, $N_r = 4$, $N_\alpha = 5$. The SMF methods were trained for each scene using a similar configuration to the original work [5]. The Arc-LSTM-SMF method was also trained separately for each scene. While not being strictly necessary, we found that it helped the model better capture scene-specific cues. We used Adam [41] optimizer with an initial learning rate of $\alpha = 0.001$, with a batch size of 8 and the number of training epochs exceeding 100. The model is open-source and is available at https://github.com/pedro-mgb/pedestrian-arc-lstm-smf.

TABLE I

INFORMATION ABOUT THE DATASET BEING USED. TOTAL OF FOUR SCENES, WITH TRAJECTORIES OF LENGTH $L = 21$ FOLLOWING THE TRAJNET++ CONFIGURATION [25]. THE NUMBER OF RELEVANT TRAJECTORIES IS THE NUMBER OF PRIMARY PEDESTRIANS. TO GET A NOTION OF THE CROWD DENSITY PER SCENE, THE AVERAGE NUMBER OF NEIGHBOURING PEDESTRIANS IS SHOWN. EACH CELL HAS TWO VALUES: THE FIRST REGARDS THE TRAINING SET, AND THE SECOND THE TEST SET.

Scene	Original no. of trajectories	No. of primary pedestrians	Avg. no. of neighbours
ETH [9]	194/124	34/3	23/10
Hotel [9]	103/150	25/26	13/10
Univ [26]	435/362	624/532	66/54
Zara [26]	191/204	201/243	12/15
Total	**923/840**	**884/809**	**50/40**

4. EXPERIMENTS

4.1. Datasets

We evaluate our results on two publicly available datasets: BIWI Walking Pedestrians dataset [9] (commonly known as ETH) and Crowds by example dataset [26] (commonly known as UCY). BIWI dataset has two scenes: ETH and Hotel. Crowds dataset also has two scenes: Univ and Zara (although Zara is often divided in Zara1 and Zara2 [10], [11], the actual location is the same). However, the original dataset trajectories were not used, because they often contained static and linear trajectories, which usually have less influencing cues (scene or social) to be captured. As such, we adopt the Trajnet++ configuration [25] for these datasets. We divided each of the four scenes in half, having around 50% of trajectories for training, and 50% for testing, with fixed trajectory length $L = 21$. The trajectories were converted to the Trajnet++ format using publicly available code[1]. In Trajnet++, instead of evaluating with all trajectories, only the primary pedestrians are evaluated. For a set of trajectories, there is only one primary pedestrian, which is the one whose trajectory is richer from the point of view of social interactions. A summary of the training and testing sets can be seen in table I, with number of original trajectories, the resulting number of primary pedestrians, and average number of neighbours, to get an idea of crowd density.

The scene with the highest number of pedestrians and overall crowd density is Univ. It has more primary pedestrians than the original number of trajectories, meaning that for the same original trajectory, more than one primary pedestrian, or in other words, more than one portion of length $L = 21$ can be retrieved. The ETH and Hotel scenes have a much smaller number of primary pedestrians, due to having more static and linear trajectories and overall smaller duration.

4.2. Evaluation metrics

To properly evaluate the performance of our proposed Arc-LSTM-SMF model, we need a diverse set of evaluation metrics. First, we use geometrical metrics (computed on primary pedestrians), a popular choice in related works [7]:

[1] https://github.com/vita-epfl/trajnetplusplusdataset; use of *chunk_stride*=21

- **Average Displacement Error (ADE)**: Average Euclidean distance between GT and model prediction over all predicted time steps $t \in \{T_{obs}+1, ..., T_{pred}\}$.
- **Final Displacement Error (FDE)**: Euclidean distance between GT and model prediction for the final prediction instant $t = T_{pred}$.

The lower these metrics are, the closer the prediction is to the GT. However, they give no real insight on how social and scene-specific cues are being followed. We use the following interaction-centric metrics, as proposed in the original Trajnet++ benchmark [25]:

- **Prediction collision (Col-P)**: The percentage of collisions between the primary pedestrian and his/her neighbours. A collision is set to occur if two pedestrians are below a safety distance T. This metric uses the predicted trajectories for primary pedestrians and neighbours.
- **GT collision (Col-GT)**: The percentage of collisions between the primary pedestrian and his/her neighbours, but using the GT trajectories for the neighbours.

Lower percentages of collisions mean the model better learns the concept of collision avoidance. This also means that the generated trajectories are more socially acceptable. We use $T = 0.1\,\mathrm{m}$, the same value used in Trajnet++ [25].

For scene-compliant evaluation, we developed new metrics that, to our knowledge, have never been employed in the in the ETH/UCY datasets. We built a simple map for each scene identifying obstacles and unwalkable areas (with the exception Univ scene, that has no actual obstacles), as well as scene limits. These new metrics are based on the maps:

- **Collisions with Scene Environment (CSE)**: Percentage of trajectories that collide with an obstacle or go to an unwalkable region. Such collisions occur if the predicted trajectory intersects the obstacle or the limits of that region, defined in the map of each scene.
- **Out of Scene Bounds (OSB)**: Percentage of trajectories that go out of the bounds of a scene. A predicted trajectory goes out of scene bounds if it goes beyond the limits set in the map specific to that scene. All train and test set trajectories are within those limits.

To get extra data for CSE and OSB, they will be computed on primary pedestrians and neighbours. The Trajnet++ configuration is interaction-centric, but not scene-centric, so scene environment cues can have relevance for some neighbour trajectories.

4.3. Baselines

We compare our work against the following baselines:

1) Constant Velocity (CV): Simple method that uses a constant velocity equal to the last observed velocity.
2) SMF [5]: Deterministic and scene-specific SMF method.
3) S-LSTM [10]: Social LSTM model, which considers interactions via a grid-based social pooling layer.
4) D-LSTM [25]: Directional LSTM model that uses a grid-shaped pooling of neighbour's relative velocity.
5) S-GAN [11]: Social GAN model, trained without variety loss (deterministic version)[2].

It is worth noting that none of these works integrate scene and social cues (they integrate either one or the other). Most of the methods that integrate both cues do not have public implementation, and those that do [4], [15] require considerable effort to make them support Trajnet++ data.

4.4. Quantitative results

The first step is to evaluate the overall quality of the Arc-LSTM-SMF predictions. The results for ADE and FDE metrics can be seen in table II.

The Arc-LSTM-SMF model has competitive ADE and FDE values with state-of-the-art models, outperforming models like D-LSTM and S-GAN. The model has considerably lower errors in the ETH and Hotel scenes, due to being specialized in each scene and having SMF predictions. The model with the second best performance is S-LSTM. D-LSTM has a larger error than S-LSTM, which is consistent with the original Trajnet++ results [25]. The CV method, while being the most simple, has the third lowest ADE. The performance of CV has already been discussed recently [42]. Geometrically, the SMF has the worst performance of all models. Even though SMF predictions are worse on their own, they give additional information to the LSTM - learning when to use it and when not -, which allows improvement of its own predictions.

To evaluate the compliance of scene and social cues, we perform evaluation with the same models and data, and using the Col-P/Col-GT (social) and CSE/OSB (scene) metrics. The results are summarized in table III.

In terms of social metrics, the Arc-LSTM-SMF continues to be competitive with the state-of-the-art, having the lowest Col-P percentage and the second lowest Col-GT percentage. This shows that the use of FOV pooling can be a superior approach in incorporating social interactions than grid-based pooling like that of S-LSTM and D-LSTM. In terms of scene-specific metrics, the model has the lowest value of CSE, along with SMF, meaning the integration of SMF with LSTM networks helps reduce the number of predictions that are not compliant with the scene environment. However, our Arc-LSTM-SMF model fails to learn the concept of scene bounds. Even though it has access to SMF predictions, which are within the scene bounds (they are inside the original video image), it cannot restrict its own predictions to the scene bounds. More insight on this will be given in the next section.

4.5. Qualitative results

This section provides a different insight on the Arc-LSTM-SMF performance, when compared to the baselines. To do that, we visualise several predictions from our model and some baselines, while also showing the GT, to understand how accurate are each of the predictions. The predictions to be shown are a small but relevant sample, containing distinct situations to highlight the strengths and limitations of our Arc-LSTM-SMF model.

[2]It was experimentally found that this model yielded better results in a unimodal evaluation than the one with variety loss.

TABLE II

COMPARING GEOMETRIC ERRORS BETWEEN ARC-LSTM-SMF MODEL AND SEVERAL BASELINES. EACH CELL HAS THE ADE, FOLLOWED BY THE FDE, BOTH IN *metres*. "AVERAGE" IS THE AVERAGE OF THE ERRORS FROM THE 4 SCENES (SAME WEIGHT FOR EACH SCENE). "WEIGHTED AVERAGE" IS THE AVERAGE OF ALL ERRORS, AND SCENES WITH MORE TRAJECTORIES HAVE MORE WEIGHT. LOWEST ERROR (BEST MODEL) IN **BOLD AND UNDERLINE**.

Scene	CV	SMF [5]	S-LSTM [10]	D-LSTM [25]	S-GAN [11]	Arc-LSTM-SMF **(ours)**
ETH	0.73/1.04	0.97/**0.72**	1.18/1.74	1.14/1.69	0.75/0.91	**0.67**/0.73
Hotel	0.52/1.03	0.75/1.41	0.50/0.95	0.62/1.26	**0.47**/0.93	**0.47**/**0.79**
Univ	0.68/1.49	1.30/2.41	0.69/1.47	0.74/1.60	0.73/1.49	**0.67**/**1.44**
Zara	0.53/1.21	0.95/1.67	**0.49**/**1.07**	0.50/1.09	0.55/1.14	0.54/1.13
Average	0.62/1.19	0.99/1.55	0.71/1.31	0.75/1.41	0.62/1.12	**0.59**/**1.02**
Weighted Average	0.63/1.38	1.17/2.14	**0.62**/1.33	0.66/1.44	0.67/1.36	**0.62**/**1.32**

TABLE III

COMPARING SOCIAL AND SCENE COMPLIANCE OF OUR ARC-LSTM-SMF MODEL, ALONG WITH SEVERAL BASELINES. THE SOCIAL METRICS (COL-P AND COL-GT) WERE COMPUTED ONLY ON THE PRIMARY PEDESTRIANS, AND THE SCENE METRICS (CSE AND OSB) WERE COMPUTED ON BOTH PRIMARY PEDESTRIANS AND NEIGHBOURS. ALL VALUES ARE SHOWN IN PERCENTAGE (%). LOWEST VALUE (BEST MODEL) IN **BOLD AND UNDERLINE**.

Metrics	CV	SMF [5]	S-LSTM [10]	D-LSTM [25]	S-GAN [11]	Arc-LSTM-SMF **(ours)**
Col-P / Col-GT (%)	11.1/11.6	10.8/14.5	10.3/**10.0**	7.7/11.4	12.7/11.3	**7.3**/10.1
CSE / OSB (%)	1.5/14.5	**0.4**/**0**	1.1/10.9	1.1/12.9	0.9/6.0	**0.4**/12.2

First, the scene-compliance of the Arc-LSTM-SMF is qualitatively evaluated. Its predictions are compared to the scene-specific SMF - which has very good performance in CSE and OSB metrics -, as well as a simple CV and the S-GAN method (do not consider the scene). A total of three situations are shown in fig. 4. The scene environment poses more weight for these situations than the social interactions, and as such the neighbours are omitted from the visualisations. The maps built for the CSE and OSB metrics are also shown in the figure, to get a visual idea of how the scenes are structured.

The first situation, in fig. 4a, is fairly simple. The Arc-LSTM-SMF, having access to scene-specific SMF predictions, was able to predict the most accurate speed and direction of motion. The simple CV baseline generated an inaccurate prediction, going into an unwalkable region, and then out of scene bounds. Even though CV has reasonably small displacement errors on average, it can quite easily generate predictions that are not compliant with the scene environment. For the second situation (fig. 4b), both CV and S-GAN generate a prediction that collides with an obstacle. The Arc-LSTM-SMF has the most accurate prediction, while also avoiding the obstacle (although narrowly). The SMF prediction also avoids the obstacle, but it is much more deviated from the GT. It is worth mentioning that the obstacle with which CV and S-GAN predictions collide with, is a set of cars that are parked. While they are present in all train and test set trajectories, they cannot be considered an entirely static obstacle. In a real-world application, the Arc-LSTM-SMF should be able to handle changes in the environment, particularly regarding scene-specific elements. A way to tackle this would be to make the Arc-LSTM-SMF model support online learning, *i.e.*, to learn how the environment evolves as new trajectories are seen or extracted. The situation from fig. 4c shows a limitation of the Arc-LSTM-SMF model - inability to learn the concept of scene bounds. While SMF and S-GAN restrict the predictions to the bounds of the scene, the Arc-LSTM-SMF is unable to do this, having a similar prediction to the CV method.

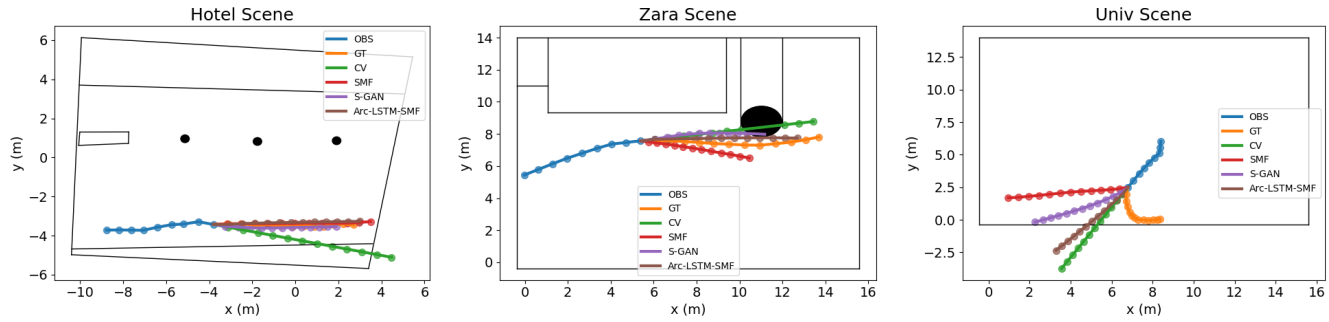
The next step is to see how socially accurate our model is when compared to the state-of-the-art, consisting of the three socially-aware methods used in section 4-D: S-LSTM, D-LSTM, and S-GAN. Four situations are shown in fig. 5. These situations include the motion of all neighbours in the scene (gray arrows), to understand how each model behaves with the surrounding social context.

The first situation, from Univ scene, in figs. 5a to 5c, highlights an advantage of the Arc-LSTM-SMF model: only considering the neighbours in front of the pedestrian. The other methods consider neighbours behind, that do not influence the pedestrian's motion. The Arc-LSTM-SMF prediction does not collide with a neighbour while other methods' do (fig. 5b). In a less crowded situation in figs. 5d to 5f, the Arc-LSTM-SMF is able to avoid some stationary neighbours (it is actually a behaviour that Arc-LSTM-SMF is commonly capable of), while other methods predict a straight trajectory (fig. 5e). There are situations where the Arc-LSTM-SMF fails to generate accurate predictions. Such is the case in figs. 5g to 5i. While the direction of the Arc-LSTM-SMF is the most accurate (fig. 5h), it cannot accurately predict the pedestrian's acceleration. The fourth trajectory, from figs 5j to 5l, shows that the Arc-LSTM-SMF model can also be socially inaccurate. The Arc-LSTM-SMF does not properly consider the neighbours moving towards the pedestrian (fig. 5j), and as such the predicted trajectory collides with a neighbour (fig. 5k).

The results from fig. 5 show that the Arc-LSTM-SMF model has the ability to generated socially accurate trajectories. However, it has still limitations regarding collision avoidance. Part of the reason for that could be that our model is not trained to directly avoid collisions. A possible area of future research involves creating an auxiliary training scheme to minimize collisions between pedestrians, *e.g.*, based on a metric like Col-P or Col-GT.

5. CONCLUSIONS AND FUTURE WORK

Pedestrian motion can be influenced by many factors. We highlighted two types of factors - presence of obstacles and



(a) (b) (c)

Fig. 4. Examples of 3 situations to highlight the scene compliance of 4 models. Best viewed in colour. The models are: CV (green), SMF (red), S-GAN (purple), and our Arc-LSTM-SMF (brown). Past trajectory (OBS) and real future trajectory of primary pedestrian (GT) in blue and orange, respectively. **(a)** Hotel scene. The Arc-LSTM-SMF avoids colliding with a building. **(b)** Zara scene. The Arc-LSTM-SMF is able to not collide with an obstacle, while other baselines collide. **(c)** Univ scene. The Arc-LSTM-SMF is not able to restrict its prediction to the bounds of the scene.

social interactions between pedestrians. The proposed Arc-LSTM-SMF considers these two factors into its predictions. To our knowledge, it is the first to do this using only pedestrian trajectories. It does not require extra information such as semantic maps or video frames for scene-complaint trajectory forecast.

The experimental results show that our Arc-LSTM-SMF model is able to outperform several state-of-the-art methods in the task of trajectory forecasting. The number of collisions with obstacles has been reduced, and it has competitive results in terms of collision avoidance between pedestrians. Nonetheless, the model still has some limitations. It cannot restrict the trajectories to the scene bounds, and cannot cope with changes in the scene environment.

Future work should dive into online training, to learn changes in the environment as new trajectories are seen or extracted. To improve the social accuracy of our model, we should explore direct training to enforce social concepts like collision avoidance.

References

[1] W.-C. Ma, D.-A. Huang, N. Lee, and K. Kitani, "Forecasting interactive dynamics of pedestrians with fictitious play," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 4636–4644.

[2] A. Sadeghian, F. Legros, M. Voisin, R. Vesel, A. Alahi, and S. Savarese, "Car-net: Clairvoyant attentive recurrent network," in *European Conference on Computer Vision (ECCV)*, 2018, pp. 151–167.

[3] R. Mehran, A. Oyama, and M. Shah, "Abnormal crowd behavior detection using social force model," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2009, pp. 935–942.

[4] J. Liang, L. Jiang, J. C. Niebles, A. G. Hauptmann, and L. Fei-Fei, "Peeking into the future: Predicting future person activities and locations in videos," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 5725–5734.

[5] C. Barata, J. Nascimento, J. Lemos, and J. Marques, "Sparse motion fields for trajectory prediction," in *Pattern Recognition*, vol. 110, 2021.

[6] D. Helbing and P. Molnár, "Social force model for pedestrian dynamics," in *Physical Review E*, vol. 51, no. 5, 1995, pp. 4282–4286.

[7] A. Rudenko, L. Palmieri, M. Herman, K. M. Kitani, D. M. Gavrila, and K. O. Arras, "Human motion trajectory prediction: a survey," in *The International Journal of Robotics Research (IJRR)*, vol. 39, no. 8, Jun 2020, p. 895–935.

[8] K. Yamaguchi, A. C. Berg, L. E. Ortiz, and T. L. Berg, "Who are you with and where are you going?" in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011, pp. 1345–1352.

[9] S. Pellegrini, A. Ess, K. Schindler, and L. van Gool, "You'll never walk alone: Modeling social behavior for multi-target tracking," in *IEEE International Conference on Computer Vision (ICCV)*, 2009, pp. 261–268.

[10] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social lstm: Human trajectory prediction in crowded spaces," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 961–971.

[11] A. Gupta, J. Johnson, L. Fei-Fei, S. Savarese, and A. Alahi, "Social gan: Socially acceptable trajectories with generative adversarial networks," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 2255–2264.

[12] D. Ridel, N. Deo, D. Wolf, and M. Trivedi, "Scene compliant trajectory forecast with agent-centric spatio-temporal grids," in *IEEE Robotics and Automation Letters (RA-L)*, vol. 5, no. 2, 2020, pp. 2816–2823.

[13] A. Sadeghian, V. Kosaraju, A. Sadeghian, N. Hirose, H. Rezatofighi, and S. Savarese, "Sophie: An attentive gan for predicting paths compliant to social and physical constraints," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 1349–1358.

[14] A. Mohamed, K. Qian, M. Elhoseiny, and C. Claudel, "Social-stgcnn: A social spatio-temporal graph convolutional neural network for human trajectory prediction," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 14 412–14 420.

[15] T. Salzmann, B. Ivanovic, P. Chakravarty, and M. Pavone, "Trajectron++: Multi-agent generative trajectory forecasting with heterogeneous data for control," in *European Conference on Computer Vision (ECCV)*, 2020, pp. 683–700.

[16] M. Karimzadeh, F. Gerber, Z. Zhao, and T. Braun, "Pedestrians trajectory prediction in urban environments," in *International Conference on Networked Systems (NetSys)*, 2019, pp. 1–8.

[17] L. Sun, Z. Yan, S. M. Mellado, M. Hanheide, and T. Duckett, "3dof pedestrian trajectory prediction learned from long-term autonomous mobile robot deployment data," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 5942–5948.

[18] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *International Conference on Learning Representations (ICLR)*, 09 2014.

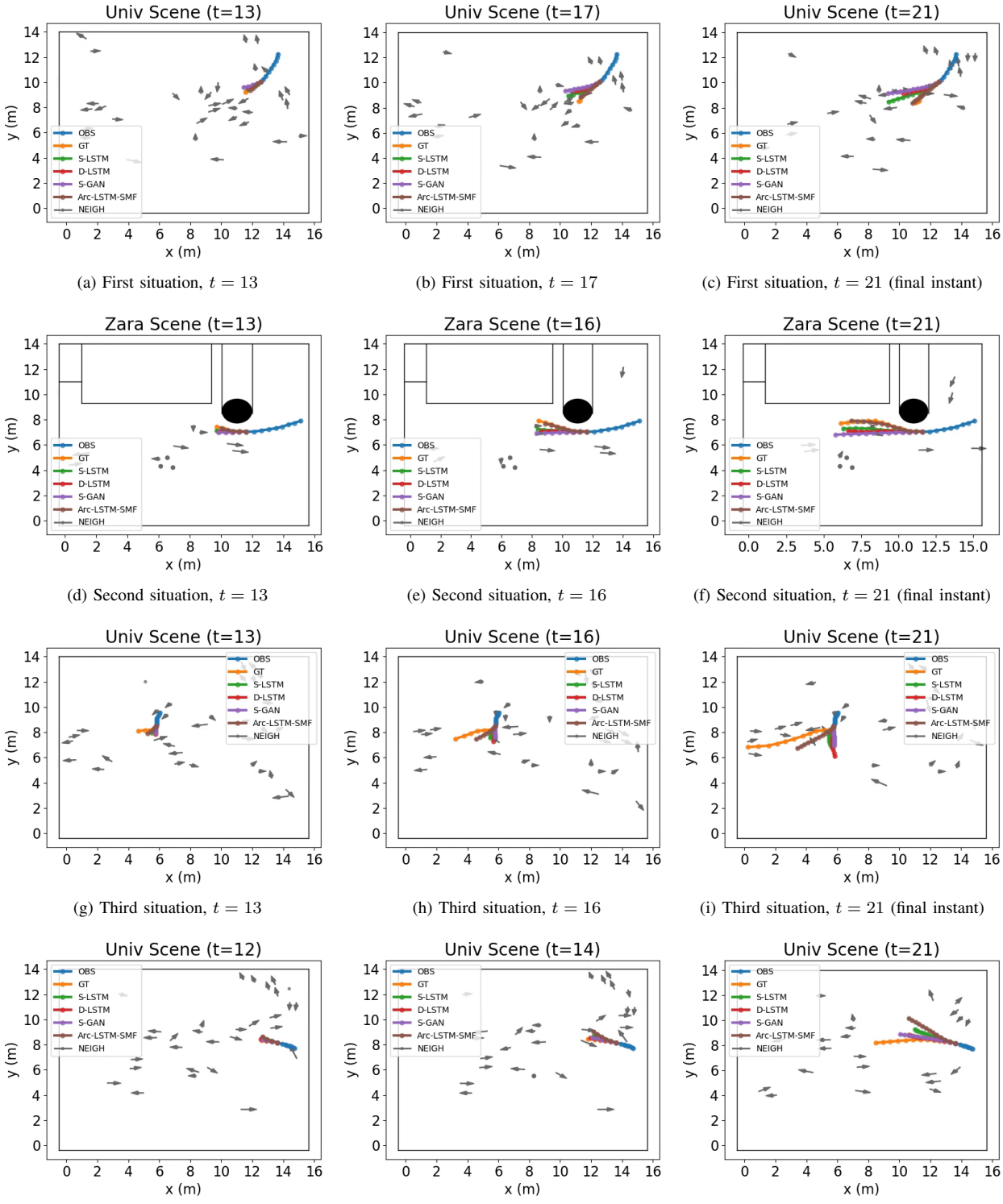
[19] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Lecture Notes in Computer Science (LNCS)*, vol. 9351, 10 2015, pp. 234–241.

[20] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[21] S. Hochreiter and J. Schmidhuber, "Long short-term memory," in *Neural Computation*, vol. 9, no. 8, 1997, pp. 1735–1780.

[22] F. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with lstm," in *Neural Computation*, vol. 12, 10 2000, pp. 2451–71.

[23] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition," in *arXiv preprint*, 2014. [Online]. Available: https://arxiv.org/pdf/1402.1128.pdf



(a) First situation, $t = 13$ (b) First situation, $t = 17$ (c) First situation, $t = 21$ (final instant)

(d) Second situation, $t = 13$ (e) Second situation, $t = 16$ (f) Second situation, $t = 21$ (final instant)

(g) Third situation, $t = 13$ (h) Third situation, $t = 16$ (i) Third situation, $t = 21$ (final instant)

(j) Fourth situation, $t = 13$ (k) Third situation, $t = 16$ (l) Third situation, $t = 21$ (final instant)

Fig. 5. Examples of 4 social situations. Best viewed in colour. There are predictions of 4 models: S-LSTM in green, D-LSTM in red, S-GAN in purple, and our Arc-LSTM-SMF in brown. Past trajectory (OBS) and GT of primary pedestrian (GT) in blue and orange, respectively. The GT neighbour motion (NEIGH) is shown in gray. **(a), (b), (c)** Situation from Univ scene. Arc-LSTM-SMF avoid collision with neighbour. **(d), (e), (f)** Situation from Zara scene. Arc-LSTM-SMF avoids stationary neighbour, while other methods do not. **(g), (h), (i)** Situation from Univ scene. Arc-LSTM-SMF avoids collision, but does not predict accurate speed. **(j), (k), (l)** Situation from Univ scene. Arc-LSTM-SMF prediction collides with neighbour, while other predictions maintain some distance from the neighbour.

[24] J. C. Nascimento, M. A. T. Figueiredo, and J. S. Marques, "Activity recognition using a mixture of vector fields," in *IEEE Transactions on Image Processing*, vol. 22, no. 5, 2013, pp. 1712–1725.

[25] P. Kothari, S. Kreiss, and A. Alahi, "Human trajectory forecasting in crowds: A deep learning perspective," in *IEEE Transactions on Intelligent Transportation Systems (ITS)*, 2021.

[26] A. Lerner, Y. Chrysanthou, and D. Lischinski, "Crowds by example," in *Computer Graphics Forum (CVF)*, vol. 26, 09 2007, pp. 655–664.

[27] L. Ballan, F. Castaldo, A. Alahi, F. Palmieri, and S. Savarese, "Knowledge transfer for scene-specific motion prediction," in *Lecture Notes in Computer Science (LNCS)*, vol. 9905, 10 2016, pp. 697–713.

[28] E. Shelhamer, J. Long, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 39, no. 4, 2017, pp. 640–651.

[29] C. Blaiotta, "Learning generative socially aware models of pedestrian motion," in *IEEE Robotics and Automation Letters (RA-L)*, vol. 4, no. 4, 07 2019, pp. 3433–3440.

[30] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems (NIPS)*, vol. 27, 2014, pp. 2672–2680.

[31] M. Mirza and S. Osindero, "Conditional generative adversarial nets," 11 2014. [Online]. Available: https://arxiv.org/abs/1411.1784.pdf

[32] J. Amirian, J. Hayet, and J. Pettré, "Social ways: Learning multi-modal distributions of pedestrian trajectories with gans," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2019, pp. 2964–2972.

[33] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "Infogan: Interpretable representation learning by information maximizing generative adversarial nets," in *Advances in Neural Information Processing Systems (NIPS)*, vol. 29, 2016, pp. 2172–2180.

[34] T. Zhao, Y. Xu, M. Monfort, W. Choi, C. Baker, Y. Zhao, Y. Wang, and Y. N. Wu, "Multi-agent tensor fusion for contextual trajectory prediction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019, pp. 12 118–12 126.

[35] V. Kosaraju, A. Sadeghian, R. Martín-Martín, I. Reid, H. Rezatofighi, and S. Savarese, "Social-bigat: Multimodal trajectory forecasting using bicycle-gan and graph attention networks," in *Advances in Neural Information Processing Systems*, vol. 32, 2019, pp. 137–146.

[36] F. Bartoli, G. Lisanti, L. Ballan, and A. Del Bimbo, "Context-aware trajectory prediction," in *24th International Conference on Pattern Recognition (ICPR)*, 2018, pp. 1941–1946.

[37] H. Xue, D. Q. Huynh, and M. Reynolds, "Ss-lstm: A hierarchical lstm model for pedestrian trajectory prediction," in *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2018, pp. 1186–1194.

[38] A. Syed and B. T. Morris, "Sseg-lstm: Semantic scene segmentation for trajectory prediction," in *2019 IEEE Intelligent Vehicles Symposium (IV)*, 2019, pp. 2504–2509.

[39] G. J. McLachlan and T. Krishnan, *The EM algorithm and extensions*. John Wiley & Sons, 2007, vol. 382.

[40] I. Hasan, F. Setti, T. Tsesmelis, A. Del Bue, F. Galasso, and M. Cristani, "Mx-lstm: Mixing tracklets and vislets to jointly forecast trajectories and head poses," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 6067–6076.

[41] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *arXiv preprint*, 2014. [Online]. Available: https://arxiv.org/pdf/1412.6980.pdf

[42] C. Schöller, V. Aravantinos, F. Lay, and A. Knoll, "What the constant velocity model can teach us about pedestrian motion prediction," in *IEEE Robotics and Automation Letters*, vol. 5, no. 2, 2020, pp. 1696–1703.